International Journal of Advances in Engineering and Management (IJAEM)
Volume 2, Issue 4, pp: 661-667 www.ijaem.net ISSN: 2395-5252

32k High-Precision Fixed Point-Fft Processor

[1]Kiran Kumar komarapuri, [2]Prasad T, [3]Subramanyeswara Rao T.J.V
[1]*PG Student, Sasi Institute of Technology & Engineering, Tadepalligudem*
[2]*Asst Professor, Sasi Institute of Technology & Engineering, Tadepalligudem*
[3]*HOD &Professor, Sasi Institute of Technology & Engineering, Tadepalligudem*
Corresponding Author: Kiran Kumar komarapuri

Date of Submission: 02-08-2020 Date of Acceptance: 20-08-2020

ABSTRACT: Fast Fourier transform (FFT) plays a vital role in digital signal processing systems. In this revision, explore the very large-scale integration (VLSI) design of high-precision fixed-point reconfigurable FFT processor. To achieve high accuracy under the limited word length, optimisation scheme is proposed to configure the scaling operations of the cascaded arithmetic blocks at each stage for yielding the most optimized accuracy for a precise FFT length. On the basis of this method, they further present a VLSI implementation of area-efficient and high-precision FFT processor, which can achieve power-of-two FFT from 32 to 8192 points. For various applications, a stimulating option is to expenditure algorithmic-based fault tolerance (ABFT) techniques that effort to consume the algorithmic belongings to detect and correct errors. ABFT is under studied from applications of signal processing and communication. One model is fast Fourier transforms (FFTs) that are a key building block in many systems. Several protection patternneed been proposed to perceive and correct errors in FFTs. In modern communication systems, it is ever more common to find several blocks operating in parallel. Recently, a method that exploits this fact to implement fault tolerance on parallel filters has been proposed. In this work passing, this technique is first applied to protect FFTs. Then, two higher protection schemes that come simultaneously the use of error correction codes and Parseval checks are proposed and evaluated. The results show that the future schemes can advance decrease the performance cost of protection.

Indexterms: Fast Fourier Transform (FFT), FFT Processor, VLSI,ABFT, Quatus II Tool.

I. INTRODUCTION

The fast Fourier transform (FFT) is one of the most important and necessary algorithms in the digital signal processing area. A large quantity of FFT lengths, mainly the power of two, have confirmed particularly valuable for different applications such as communication, image processing and biomedical signal processing.

During the last two many years, an expansion of architectures have been proposed for constant length FFT processors, which may be labelled into catalogues: reminiscence primarily based and pipelined. The reminiscence based building [3] provides a low-electricity solution; but, this approach suffers from long latency and might require extra buffer area for machine synchronisation. The pipelined architecture including single path delay feedback (SDF) pipeline FFT structure [1] and multipath delay commutator (MDC) structure [9], provides excessive throughputs, but it requires more hardware assets on the same time. To accomplish flexibility, most reconfigurable FFT processors reported inside the literature enhance and optimise the datapaths of the fixed size FFT processor by resources of multiplexer switching.

A variable-length FFT processor that integrates two radix-2 stages and three radix-8 stages for FFT sizes 512, 1024 and 2048 was proposed in [1, 10, 11] targeting the reconfigurable FFT processor (128–2048/1536-point) for long-turn evolution systems, where the former proposed a radix-2^3 SDF pipeline architecture enabling the 1526-point FFT computation to be compatible with the conventional power-of-two-based SDF architecture and the latter proposed an L-parallel M-point SDF pipeline FFT architecture to produce a parallel pipeline FFT processor capable of variable length FFT computations.

We analyse the fixed-point arithmetic use in FFT processor and present the statistics-based method to optimize the scaling operations of the cascaded arithmetic blocks in reconfigurable FFT processors. Inspection 4, the proposed reconfigurable FFT architecture is illustrated and the VLSI implementation and comparison results are presented in Section 5. Finally, a conclusion isFifth complexity of communications and signal processing circuits increases every year. The method achieved by the CMOS technology scaling that enables the

International Journal of Advances in Engineering and Management (IJAEM)
Volume 2, Issue 4, pp: 661-667 www.ijaem.net ISSN: 2395-5252

combinations of more and more transistors on a single device.

This greater than the complexity makes the circuits more susceptible to errors. At the same time, the scaling resource that transistors work with low voltages and are more incline to errors caused by noise and manufacturing deviations[1].The importance of induced soft errors similarlyrise as technology scales [2]. The errors can differ the logical value of a circuit node creating a temporary error that can affect the system operation. To consent to those soft errors, a wide variety of techniques can be used [3]. These include processes for the integrated circuits like, for example, the silicon on insulator. Another option is to proposed basic circuit blocks or complete design libraries to minimize the probability of soft errors. Finally, it is also possible to additional redundancy at the system level to detect and correct errors.

One classical example is the custom of triple modular redundancy (TMR) that triples a block and votes among the three outputs to detect and correct errors. For example, for TMR, the below is >200%.This is because the unprotected module is replicated three times (which requires a 200% overhead versus the unprotected module), and additionally needed to correct the errors making the overhead >200%. This overhead is excessive for many applications. Alternative approach is to use the algorithmic properties of the circuit to detect/correct errors. This is regular referred to as algorithm-based fault tolerance (ABFT) [4].

II. LITERATURE SURVEY

In [12], Kai-Jiun et al. proposed an MDC-based architecture for multiple-input–multiple-outputorthogonal frequency-division multiplexing system with variable length.

In [13], Anthony et al. explored the memory-based FFT architecture and proposed a configurable addressing scheme for a continuous-flow memory-based FFT processor. The above works have proposed various architectures for the reconfigurable FFT processors, but few of them have given a deep insight of the cost-accuracy tradeoff caused by the finite wordlength effect. As known, fixed-point arithmetic is used in the practical very large-scale integration (VLSI) implementation of FFT algorithms, where all coefficients and input signals have to be represented with a finite number of bits in binary format depending on the tradeoff between the hardware cost and the accuracy of output signals. The theoretical performance evaluation of fixed point FFT has been presented in previous works.

Wei-Hsin and Truong [14], Sarbishei and Radecka [15] and Mohammad Reza and Lesley [16] investigated the quantisation property of the radix-r or split-radix FFT module with varying operating wordlength.

jian et al. [17] further presents a fixed-point analysis and hardware evaluation of radix-2k FFT with SDF and MDC pipelined structures. However, the remaining abstract study still reveals insufficiency to manage with the problems in the applied design of reconfigurable FFT processors. To the best of our knowledge, most of the practical FFT designs instinct use rounding or truncation operations to keep the word length under the constrained bit budget, while few works, especially for variable-length FFT processors, have considered the mixed use of multiple scaling schemes to compensate the quantization noise, which motivates us to carry out the research in this paper. Therefore, to get comparable accuracy with fewer bit budgets, this explores to carries out a practical analysis of the fixed-point arithmetic blocks used in FFT implementation. Unlike conventional designs that use a unified scaling approach to keep the word length, this work proposes the mixed use of multiple scaling approaches to improve the accuracy of fixed-point FFT implementation. In accumulationof a statistics-based optimisation is proposed method to configure the scaling approaches of the cascaded arithmetic division part at each stage for yielding the most optimized accuracy for a given FFT length. As an example, a complete implementation of low-cost high-precision pipelined reconfigurable FFT processor supporting 32-8K-point FFT is further proposed. The proposed FFT processor output word length can provide an output signal-to-quantization-noise ratio (SQNR) of 53.18 dB at 8K FFT mode, and the chip area and power consumption are also like when compared with the state-of-the-art FFT implementations.

III. PROPOSED PROTECTION SCHEMES FOR PARALLEL FFTS

The initials point for our work is the protection scheme based on the use of ECCs that was presented in [17] for digital filters. This scheme is shown in Fig. 1. In this example, a single error correction Parseval check code [18] is used. The main system contains of four FFT modules and three redundant modules are added to detect and correct errors. The input through three redundant modules is combinations of the inputs and they are used to check linear combinations of the outputs. For example, through the input to the first redundant module is

$$x_5 = x_1 + x_2 + x_3 \quad (1)$$

and since the DFT is a linear operation, its output z_5 can be used to check that

$$z_5 = z_1 + z_2 + z_3. \quad (2)$$

International Journal of Advances in Engineering and Management (IJAEM)
Volume 2, Issue 4, pp: 661-667 www.ijaem.net ISSN: 2395-5252

This will be noted as c1 check. The same reason applies to the other two redundant modules that will provide check sum of c2 and c3. Based on the parameters observed on each of the checks, the ABFT module on which the error has occurred can be determined. The different patterns and the corresponding errors are summarized in Table I Once the module in error is known, the error can be corrected by rebuilds its output using the next level modules. For example, for an error may effect z1, this can be done as follows:

$$z_{1c}[n] = z_5[n] - z_2[n] - z_3[n]. \quad (3)$$

Same as the Error correction equations can be used to correct errors on the other modules. More forward delay of ECCs can be used to correct errors on secondary modules if that is needed in a given application. The overhead of this technique, as details given [17], is SOS check lower than TMR as the number of redundant FFTs is compatibles to the logarithm of the number of same FFTs. From the example, to protect six FFTs, three redundant FFTs are needed, but need to protect eight, the number of redundant FFTs in only three. This shows how the overhead greatly decreases with the number of FFTs.

ERROR LOCATION IN THE HAMMING CODE

$c_1\, c_2\, c_3$	Error Bit Position
0 0 0	No error
1 1 1	z_1
1 1 0	z_2
1 0 1	z_3
0 1 1	z_4
1 0 0	z_5
0 1 0	z_6
0 0 1	z_7

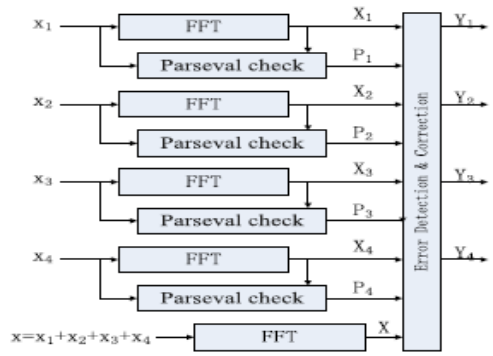

Fig 1. Parity-SOS (first technique) fault-tolerant parallel FFTs

In Section I, it has been mentioned that over the years, many authors' methodology's have been proposed to protect the FFT. One of them is the Sum of Squares (SOSs) check [5] that can be used to detect errors. The total SOS check is based on the Parseval scheme states that the SOSs of the inputs to the FFT are equal to the SOSs of the outputs of the FFT except for a scaling factor. This combination can be used to detect errors with low overhead as one multiplication is neededfor each input or output sample (three multiplications and adders for SOS per sample/unit).For parallel FFTs, the SOS check can be combined with the ECC approach to reduce the protection overhead. Since the SOS check can only detect errors, the ECC part should be able toimplement the correction. This can be done using the equivalent of asimple parity bit for all the FFTs. In addition, the SOS check is usedon each FFT to detect errors. When an error is detected, the outputof the parity FFT can be used to correct the error. This is betterexplained with an example. In Fig. 2, the first proposed scheme is illustrated for the case of four parallel FFTs. A redundant (the parity)FFT is added that has the sum of the inputs to the original FFTs as input. An SOS check is also added to each original FFT. In case an error is detected (using P1, P2, P3, P4), the correction can be doneby recomputing the FFT in error using the output of the parity FFT(X) and the rest of the FFT outputs. For example, if an error occursin the first FFT, P1 will be set and the error can be corrected bydoing

$$X_{1c} = X - X_2 - X_3 - X_4. \quad (4)$$

This combination of parity FFT and the SOS check reduces thenumber of additional FFTs to just one and may, therefore, reduce theprotection overhead. In the following, this scheme will be referred toas parity-SOS (or first proposed technique).Another possibility to combine the SOS check and theECC approach is instead of using an SOS check per FFT, use an ECC for the SOS checks. Then as in the parity-SOS scheme, anadditional parity FFT is used to correct the errors. This secondtechnique is shown in Fig. 3. The main benefit over the first parity-SOS scheme is to reduce the number of SOS checks needed. Theerror location process is the same as for the ECC scheme in Fig. 1and correction is as in the parity-SOS scheme. In the following, thisscheme will be referred to as parity-SOS-ECC (or second proposedtechnique).

International Journal of Advances in Engineering and Management (IJAEM)
Volume 2, Issue 4, pp: 661-667 www.ijaem.net ISSN: 2395-5252

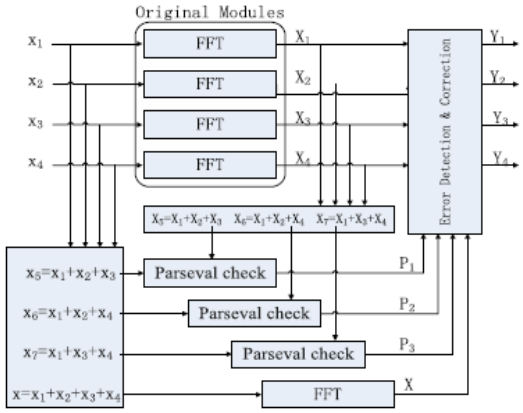

Fig 2.Parity-SOS-ECC (second technique) fault-tolerant parallel FFTs.

TABLE II
OVERHEAD OF THE DIFFERENT SCHEMES TO PROTECT *k* FFTs

	FFTs	SOS checks
ECC	$1+\log_2(k)$	0
Parity-SOS	1	k
Parity-SOS-ECC	1	$1+\log_2(k)$

The overheads of the proposed schemes can be initially estimated using the number of additional FFTs and SOS check blocks needed. This addition parity check information is summarized in Table I for benefit of k original FFT modules assumes k is a power of two modules. It can be observed that the first proposed schemes reduce the number of additional FFTs to just one. In furthermore, the second technique also reduces twiddles the number of parseval SOS checks. In Section III, a complete value for an FPGA implementation is discussed to illustrate the nearby overheads of the given proposed techniques. In all the techniques discussed, soft errors can also affect the elements added for protection. For the ECC technique, the protection schemes of these elements were discussed in [12]. In the case of the redundant module or parity FFTs, an error will have no effect as it will not circulates to the data outputs and will not trigger a correction. In the case offsets checks, an error will trigger a correction when actually there is no error on the FFT. This will cause an unnecessary correction but will also give throughout the error correct result. Finally, errors on the detection and correction blocks in Figs. 2 and 3 can propagate errors to the outputs. In our implementations, those blocks are covered with TMR. The same applies for the adders utilities to compute the inputs to the redundant module FFTs in Fig. 1 or to the SOS checks in Fig. 3. The triplication of these blocks has a small forces on circuit increase in CORDIC rotation factors as they are much simpler than the parity FFT computations' final section observation is that read CFF4 the ECC scheme can detect all check errors that exceed a given input level threshold (given by the quantization applies to implement the FFTs) [17]. On the other hand, the SOS check detects most errors but does not guarantee the detection of all errors [4]. Therefore, to compare the three techniques for a given implementation, fault additional of experiments should be proceed to determine the total percentage of errors that are actually corrected. This means that an evaluation has to be done both in terms of overhead and error coverage.

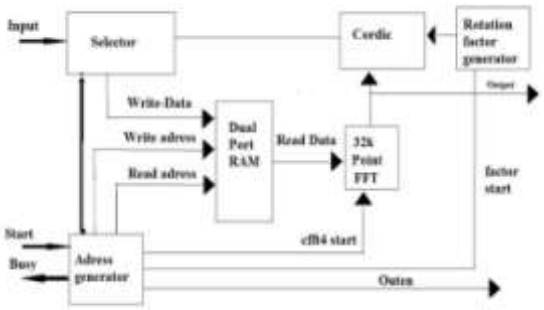

Fig 3.Architecture of the FFT implementation.

Hardware architectureoverviewFig. 3show the block diagram of the proposed FFT processor, which employs the pipelined SDF architecture. Since the high radix algorithm can save executing cycles and the number of complex multiplications, in this design, we decompose the process of 8K-point FFT into three radix-23 stages, one radix-22 stage andone mixed radix-2/22 stage. The future scheme architecture takes four complex multipliers and three,four read-only memories (ROMs) for twiddle factors. The five butterfly units use the SDF pipeline structure, where first-in–first-out (FIFO) is employed for temporary storage at each stage. Then, by means of multiplexer switching, the proposed design is able to perform 2n-point FFTcomputations starting from 32-point to 8K-point.In Cooley–Tukey algorithm, a large-point FFTcan be composed into several cascade small-point FFTs. Tomaximize the hardware utilization, we propose the pipeline configuration scheme as shown in Table 1. The 8 K/4 K/2K-pointFFT share the first four stages, but at the fifth stage, the 8K-pointFFT uses radix-4 mode, the 4K-point FFT uses radix-2 mode and the 2K-point FFT bypasses this stage. For the 1 K/512/256-pointFFT computation, the input stream ignores the first stage by multiplexing and starts from stage 2. Moreover, the 128/64/32-point FFT bypass the first two stages and start from stage 3. Length of FIFO varies as per the stage in the pipeline. The first stage has afford

International Journal of Advances in Engineering and Management (IJAEM)
Volume 2, Issue 4, pp: 661-667 www.ijaem.net ISSN: 2395-5252

size of 4096 complex samples, the second stage has a FIFOsize of 2048 complex samples, so on and so forth till the end of pipeline architecture. It is noteworthy that a significant advantage of the proposed configuration scheme is to improve the energy efficiency. In the pipelined SDF structure, the FIFOs used in the early stages are much larger than the ones used in the later stages. Thus, considering the power consumption, the early stages are much more power hungry than the later ones. The proposed configuration scheme makes use of this principle by sharing the later stages as much as possible while bypassing the early stages as long as they are trivial. Then, by simply shutting down the clock of the bypassed stages, the proposed architecture can minimize the power consumption caused by the hardware overhead. Butterfly unit as shown in Fig. 2, the proposed architecture uses three kinds of butterfly units, which are radix-23, radix-22 and mixed radix-2/22.

FFT mode	Stage 1	Stage 2	Stage 3	Stage 4	Stage 5
8K	radix-8	radix-8	radix-8	radix-4	radix-4
4K	radix-8	radix-8	radix-8	radix-4	radix-2
2K	radix-8	radix-8	radix-8	radix-4	–
1K	–	radix-8	radix-8	radix-4	radix-4
512	–	radix-8	radix-8	radix-4	radix-2
256	–	radix-8	radix-8	radix-4	–
128	–	–	radix-8	radix-4	radix-4
64	–	–	radix-8	radix-4	radix-2
32	–	–	radix-8	radix-4	–

Fig 4.area comparison of multiple synchronization models

All these butterfly units are designed using the SDF structure. Forth radix-22 and mixed radix-2/22 units, the design is straightforward. However, for the radix-23 units, non-trivial multiplication caused by W81 and W83 may incur cost and speed issues. As shown in (1), the multiplication of W81 and W83 requires at least one constant multiplication (0.707) and one addition/subtraction (1−j and −1−j). Thus, along with the addition/subtraction in butterfly operation, the multiplication of W81 and W83can is the critical path of the overall architecture. To short this critical path, the proposed architecture utilizes the feedback character of the SDF structure to optimize the micro-architecture. As shown in Fig. 3, the multiplication of W81 and W83 is broken-down into two separate parts, where the addition/subtraction for1−j and −1−j is carried out at the feedback path and the multiplication of 0.707 is carried out at the feed forward path. For the radix-2^2 and mixed radix-2/22 units, the design is straightforward. For the last stage, we propose a configurableradix-22/radix-2 butterfly unit. This butterfly architecture consists of two, three cascaded radix-2 butterflies as shown in Fig. 3. When a radix-2instead of a radix-4 computation is needed, this butterfly enables only the first internal radix-2 computations and disables the otherradix-2 computation.

To decrease the ROM storage overhead, only 2/16 period of cosine and sine simulated waveforms are stored, and the rest of the twiddle factors can be derived from quadrant conversion, which is shown in Fig. 4. For example sample index for the FFT operation, add from 0 to N−1, issued to generate this phase that addresses all the twiddle-factor ROMs. The first three most significant bits (MSB) of the inphase aroused as the control signals to produces the correct sine/cosine value from the outputs of the one-eighth-cycle tables addressed byte remaining bits of the phase. As for the requirement of variable-length FFT computation, the twiddle factors stored in the ROM are also shared by one to many FFTmodes. Taking the 2 K/4 K/8K-point FFT, for example, the twiddle factors used in 8K-point FFT can be represented apex(− j2πn/8192), where n is from 1 to 8192. When the twiddle factors used in 8K,4- and 2 K-point FFTs aertex(− j2πn/4096) = exp(− j2π2n/8192), n = 1, 2, …, 4096, and exp(− j2πn/2048) = exp(− j2π4n/8192), n = 1, 2, …, 2048,respectively. Since, the twiddle factors of 4 K/2K-point FFT are covered by the ones of 8K-point FFT. The comparisons is the step generated by the address , which changes from one in the 8K-FFT mode to two and four in 4 and 2 K FFTs, respectively. Thus, second ROM at the first stage only stores the 1/8 period of cosine and sine points of 8 K FFT. When Alter interms to the2048/4096-FFT modes, the address cordicgenerator slightly in order to the counter stepaccordingly to reuse these twiddle factors.

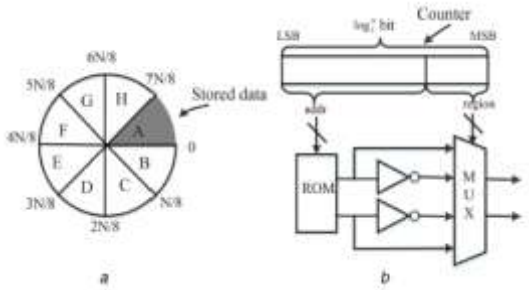

Fig 5. Indexing of twiddle factors

International Journal of Advances in Engineering and Management (IJAEM)
Volume 2, Issue 4, pp: 661-667 www.ijaem.net ISSN: 2395-5252

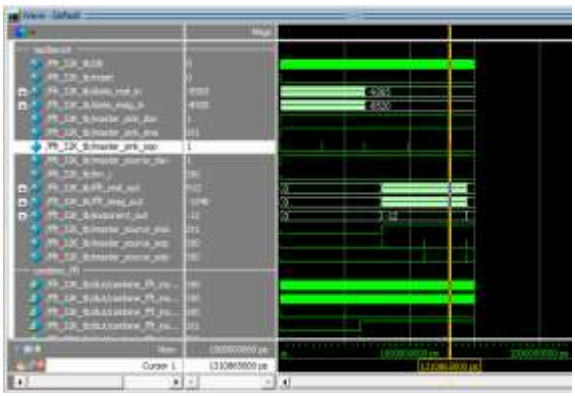

Fig 6.Simulated diagram of FFT Processor.

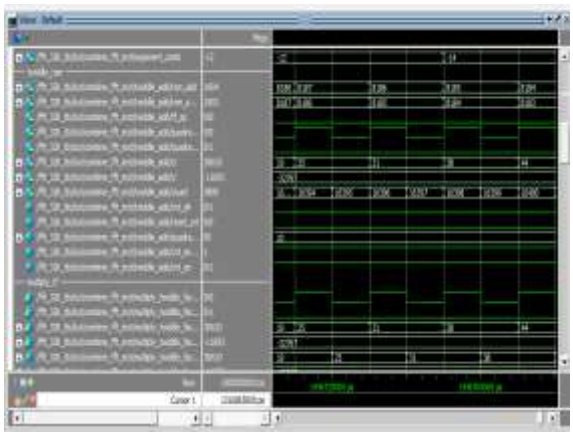

Fig 7. Simulated diagram of Twiddle factor

The results are simulated using Quartus II tool with modelsim simulator.

IV. CONCLUSION AND FUTURE WORK

In this paper, the quantization noise in the design of fixed-pontiff processor is analysed. To yield comparable accuracy with fewer bit budgets, we propose the joint use of multiple scaling approaches to improve the accuracy of practical FFTimplementation. This method of optimization scheme is proposed to explore the optimisation configuration of the cascaded arithmetic blocks at each stage for yielding the most optimisedaccuracy for various FFT configurations. By using the proposed method, a pipelined reconfigurable FFT processor supporting 32K-point FFT has been implemented in SMIC 0.13μm process. The proposed processor has a core area of 2.7 mm2, providing the highest throughput of 400 Sample/s at 400 MHz when executing the 8K-point FFT at 40 MHz, the power dissipation measured by post-layout simulation is 35.7 maws. Also, it can achieve the SQNR of 53.28 dB. Compared to the state-of-the-raft implementation, the proposed design improves the SQNRperformance, while has similar chip area and lower consumption.

In this brief, the protection of parallel FFTs implementation against soft errors has been studied. Two techniques have been proposed and evaluated. The proposed techniques are based on combining an existing ECC approach with the traditional SOS check. The SO checks are used to detect and locate the errors and a simple parityFFT is used for correction. The detection and location of the errorscan be done using an SOS check per FFT or alternatively using a setoff SOS checks that form an ECC.The proposed techniques have remain evaluated both in positions of implementation difficult and error detection capabilities. The results show that the second technique, which uses parity FFT and a set of SOS checks that form an ECC, provides the best results in terms of implementation complexity. In terms of error protection, fault injection experiments show that the ECC scheme can recover all the errors that are out of the tolerance range. The fault coverage for the parity SOS method and the parity-SOS-ECCmethod is 99.9% when the tolerance level for SOS check is 1 of the FFT.

REFERENCES

[1]. HaoXiao , Xiang Yin, Ning Wu, Xin Chen: 'VLSI design of low-cost and high-precisionfixed-point reconfigurable FFT processors, IEEE, IET Computers & Digital Techniques.,2018, pp. 1751-8601.

[2]. Kala, S., Nalesh, S., Nandy, S.K., et al.: 'Energy efficient scalable anddynamically reconfigurable FFT architecture for OFDM system'. 2014 FifthInt.Symp. Electronic System Design, Mangalore, India, December 2014, pp.20–24

[3]. Jo, B.G., Sunwoo, M.H.: 'New continuous-flow mixed-radix (CFMR) FFTprocessor using novel in-place strategy', IEEE Trans. Circuits Syst. I, Regul.Pap., 2005, 52, (5), pp. 911–919

[4]. Pei-Yun, T., Chung-Yi, L.: 'A generalized conflict-free memory addressing scheme for continuous-flow parallel-processing FFT processors with rescheduling', IEEE Trans. Very Large Scale Integr. (VLSI) Syst., 2011, 19,(12), pp. 2290–2302

[5]. Baas, B.M.: 'A low-power, high-performance, 1024-point FFT processor',IEEE J. Solid-State Circuits, 1999, 34, (3), pp. 380–387[6] Song-Nien, T., Jui-Wei, T., Tsin-Yuan, C.: 'A 2.4 GS/s FFT processor forOFDM-based WPAN applications', IEEE Trans. Circuits Syst. II, ExpressBriefs, 2010, 57

[6]. Manohar, A., Michael, B., Keshab, K.P.: 'Pipelined parallel FFT architecturesvia folding transformation', IEEE Trans. Very

Large Scale Integr. (VLSI)Syst., 2012, 20, (6), pp. 1068–1081

[7]. Mario, G., Grajal, J., Sanchez, M.A., et al.: 'Pipelined radix-2k feedforwardFFT architectures', IEEE Trans. Very Large Scale Integr. (VLSI) Syst., 2013,21, (1), pp. 23–32

[8]. Song-Nien, T., Chi-Hsiang, L., Tsin-Yuan, C.: 'An area- and energy-efficientmultimode FFT processor for WPAN/WLAN/WMAN systems', IEEE J.Solid-State Circuits, 2012, 47, (6), pp. 1419–1435

[9]. Lin, Y.T., Tsai, P.Y., Chiueh, T.D.: 'Low-power variable-length fast Fouriertransform processor', IEE Proc. Comput. Digit. Tech., 2005, 152, (4), pp.499–506

[10]. Chu, Y., Mao-Hsu, Y.: 'Area-efficient 128- to 2048/1556-point pipeline FFTprocessor for LTE and mobile WiMAX systems', IEEE Trans. Very LargeScaleIntegr. (VLSI) Syst., 2015, 23,

[11]. Anthony, T.J., Dean, N.T., Bevan, M.B.: 'The design of a reconfigurablecontinuous-flow mixed-radix FFT processor'. 2009 IEEE Int. Symp.Circuitsand Systems, Taipei, Taiwan, May 2009, pp. 1133–1136

[12]. Wei-Hsin, C., Truong, Q.N.: 'On the fixed-point accuracy analysis of FFTalgorithms', IEEE Trans. Signal Process., 2008, 56, (10), pp. 4673–4682

[13]. Sarbishei, O., Radecka, K.: 'Analysis of mean-square-error (MSE) for fixedpointFFT units'. IEEE Int. Symp. Circuits and Systems (ISCAS), Rio deJaneiro, Brazil, May 2011, pp. 1732–1735

[14]. Mohammad Reza, M., Lesley, S.: 'Minimizing the error: a study of theimplementation of an integer split-radix FFT on an FPGA for medicalimaging'. Int. Conf. Field-Programmable Technology, Seoul, Korea (South),December 2012, pp. 360–367

[15]. Jian, W., Chunlin, X., Kangli, Z., et al.: 'Fixed-point analysis and parameteroptimization of the radix-2k pipelined FFT processor', IEEE Trans. SignalProcess., 2015, 63, (18), pp. 4879–4893.